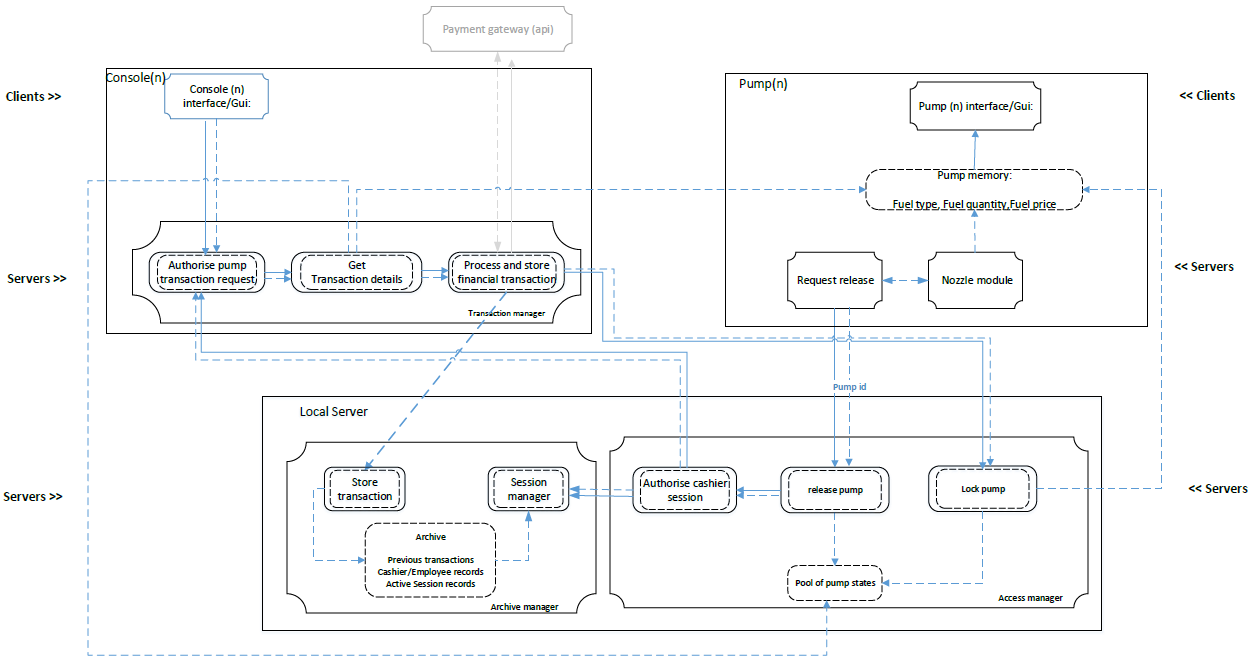
Definitions:

The diagrammatic representation below represents the interaction between a given pump (n) and a given cashier console (n). It is assumed once a console grants access the unlock request is removed from view of other consoles in the scope of unlocking a pump.

The style chosen is Client - Server. The rationale behind the choice being that the architecture can support expansion through modularity. This is a quality that would be useful for the proposed fuel station as the number of pumps is not defined.



**Components and Connector descriptions**

**Clients:**

1. Console interface: The console will receive input from physical interface devices such as a keypad and or a touch screen interface. The role of the console interface will be to allow cashiers to respond to request for pump access/activation and for processing completed transactions financially.

The console display will provide information to cashiers such as:

* A list of pumps and their statuses , inclusive of activation requests
* For any given fuel transaction information inclusive of:

Pump id

Transaction time

Fuel price per unit

Volume of fuel dispensed

Total payable, based on price per unit for the units dispensed.

# Pump Interface:

It’s will provide customers with the ability to request pump unlocks to enable them to use the pump. This however will not be an explicit action, the user will realise this function by lifting the pump nozzle. This action will automatically trigger a request to be sent to a console(s) for release.

A small lcd panel displays transaction information inclusive of:

fuel price per unit

volume of fuel dispensed

total payable , based on price per unit for the units dispensed.

Type of fuel

No customer interaction is required apart from lifting, using the nozzle and putting the nozzle back down.

# **Servers:**

# Transaction manager

3.1 Authorise pump transaction request:

Cashier response to a pump release request. Achieved through physical interaction with GUI by cashier and an automatic authorisation check.

* 1. Get transaction details

Given pump(n) has completed dispensing fuel, signified by customer placing nozzle back down, get transactions data inclusive of pump id, fuel price per unit, volume of fuel dispensed, type of fuel, and total payable based on price per unit for the units dispensed. Pump for selection determined by data in “pool of pump statuses”.

* 1. Process and store financial transaction

Using data returned from in 3.2 finalise transaction state by processing financial payment, and store transaction record in archive.

# Archive manager

4.1. Store transaction

Store a paid transaction into permanent memory on the server in the form of a database, inclusive of cashier id.

4.2. Session manager

Each cashier is required to log into their console at the start of their shift. Successful logon will start a session which expires when after a given duration unless they specifically log out.

4.3 Archive

Permanent storage containing previous transaction histories, employee details, and active session records.

# Access manager:

5.1. Authorize cashier session:

Given that a cashier has already logged into a console. An active session number will be present and the archive manager will reference this to verify whether the console is authorised to carry out an action.

5.2. Release pump

After receiving a release request from pump, send request to console and wait for a response. On response arrival check session information and proceed is valid.

5.3 Lock pump

Return pump state to locked and clear pump memory from previous transaction, ready for the next transaction. Display should reset and show default values.

# Request release

Receives input from nozzle module when nozzle is lifted and sends a request to the access manager.

# Nozzle module

On-board module that reads sensor information on whether the nozzle is in it’s default location, i.e. in pump, or has been lifted and sends a data to “Request release” if lifted.

# Pump memory

Temporary memory responsible for keeping information from the pump nozzle and information derived from the pump nozzle information, to be display by the Pump display.

**DESIGN EVALUATION:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Scenario | Change |  |  |  |
| No. | Description | Type | Component | Change |
| 1 | User wants to add the facility to charge electric vehicles. | In-Direct | Pump memory | Amount of electricity used to be added. Pump lock algorithm would need to be adapted such that a pump can be locked in use – due to the lengthy recharge process of electric vehicles. |
| 2 | User wants to add a new pump from a different manufacturer. | Direct | - | None – Under this architecture the pumps can be added in a modular fashion, provided the method of interfacing and data access and transmission remains the similar. |
| 3 | User wants a new interface for the cashier with touch screen | Direct | Console(n) interface | None – proposed system requires I/O streams and event such as clicks. The source and generation does not impact system. |
| 4 | User wants to accept payment via an NFC device | Direct | Process and store financial transaction | None - new physical hardware will still provide the same information required to complete a transaction. |
| 5 | User wants to add pay at pump facility | In-Direct | Pump interface (gui) | Physical card reader will need to be added to pump and a secure connection required to access external payment gateways. |
| 6 | User wants to add an additional grade of fuel | In-Direct | Archive | Changes to data, however the system remains semantically correct. |
| 7 | User wants to add another pump | Direct | - | None – another pump may be added modularly. |
| 8 | User want to add another cashier console | Direct | - | None – another pump may be added modularly. |
| 9 | User want to add the ability to combine multiple pump usages into a single transaction | In-Direct | get transaction details, process and store financial details, | Get transaction details must changed to obtain transaction of n pumps and not process payment prior to all pump data being received.  Store financial details needs to be modified to not associate a single transaction to a single pump id |
|  |  |  |  |  |

**I am missing the diagram ... but will work on it when I finish work.**